1. **Агрегирующие функции** в SQL предназначены для обработки данных в наборах строк и возврата единственного значения. Они выполняют вычисления на группах строк, например, суммируют числа, находят среднее значение или определяют максимальное/минимальное значение. Примеры агрегирующих функций: SUM, AVG, MAX, MIN, COUNT.
2. **SUM**: Складывает числовые значения в столбце. **AVG**: Вычисляет среднее значение числового столбца. **MAX**: Возвращает максимальное значение в столбце. **MIN**: Возвращает минимальное значение в столбце. **COUNT**: Подсчитывает количество строк в наборе данных. Названия функций отражают их действие. Например, SUM выполняет суммирование, AVG вычисляет среднее, MAX и MIN находят максимальное и минимальное значения соответственно, а COUNT подсчитывает строки.
3. Для объединения двух запросов необходимо выполнить следующие требования:
   * Оба запроса должны иметь одинаковое количество столбцов.
   * Столбцы, по которым происходит объединение, должны иметь совместимые типы данных.
   * Запросы должны быть объединены с помощью оператора UNION, INTERSECT или EXCEPT.
4. **Объединение**, **пересечение** и **разность** двух таблиц в SQL можно выполнить следующим образом:
   * **Объединение (UNION)**: Комбинирует строки из двух таблиц в один результат. Дубликаты удаляются, если используется UNION, а не UNION ALL.
   * **Пересечение (INTERSECT)**: Возвращает только общие строки, которые присутствуют в обеих таблицах.
   * **Разность (EXCEPT)**: Возвращает строки из первой таблицы, которых нет во второй таблице.
5. **Особенности применения предложения UNION**:
   * UNION объединяет результаты запросов, удаляя дубликаты.
   * Столбцы в обоих запросах должны иметь одинаковое количество и совместимые типы данных.
   * Порядок столбцов в результирующем наборе данных соответствует порядку столбцов в первом запросе.
6. В **PostgreSQL** реализованы следующие виды **JOIN**:
   * **INNER JOIN**: Возвращает только строки, которые имеют соответствующие значения в обеих таблицах.
   * **LEFT JOIN (OUTER JOIN)**: Возвращает все строки из левой таблицы и соответствующие строки из правой таблицы.
   * **RIGHT JOIN (OUTER JOIN)**: Возвращает все строки из правой таблицы и соответствующие строки из левой таблицы.
   * **FULL OUTER JOIN**: Возвращает все строки из обеих таблиц, дополняя недостающие значения NULL.
7. **Особенности применения предложения JOIN**:
   * **JOIN** используется для объединения данных из нескольких таблиц.
   * При использовании **OUTER JOIN** следует быть внимательным к порядку таблиц в запросе.
   * Необходимо указать условие объединения (через ON или USING).
8. Конструкция **WITH** (также известная как **Common Table Expression** или **CTE**) **предназначена** для создания временных результатов, которые можно использовать в последующих частях запроса. Она позволяет именовать подзапросы и делает код более читаемым и модульным. Преимущества использования **WITH**:
   * **Многократное использование**: Вы можете определить временную таблицу с помощью **WITH**, а **затем** ссылаться на нее несколько раз в запросе.
   * **Рекурсия**: **WITH** позволяет создавать рекурсивные запросы, что полезно, например, при работе с иерархическими данными.